UCONN
UNIVERSITY OF CONNECTICUT

Instructional Videos and Interactive Notebooks for Learning of Coding Concepts in Chemical Engineering Analysis

Matthew Wilhelm, PhD Candidate

Chenyu Wang, PhD Candidate

Matthew Stuber, Assistant Professor

20 AIChE
Annual Meeting, San Francisco, CA

Process Systems and Operations Research Laboratory

Challenges: Content

- ✓ **Show students how to formulate and solve engineering problems**
- ✓ **Introduce concepts of algorithms**
- ✓ **Introduce numerical methods**
- ✓ **Teaching a "new" programming language**

General CHEG Curriculum

FRESHMAN YEAR

First Semester	*Credits*	*Second Semester*	*Credits*
CHEM 1127Q General Chemistry	4	CHEM 1128Q General Chemistry	4
MATH 1131Q Calculus I	4	MATH 1132Q Calculus II	4
ENGR 1000 Orientation to Engineering	1	ENGR 1166 Foundations of Engineering	3
CSE 1010C Intro to Computing	3	Arts & Humanities (Content Area 1) [1]	3
ENGL 1010 or 1011 Academic Writing	4	Social Sciences (Content Area 2) [1]	3
	16		**17**

SOPHOMORE YEAR

First Semester	*Credits*	*Second Semester*	*Credits*
PHYS 1501Q Eng Physics I	4	PHYS 1502Q Eng Physics II	4
CHEM 2443 Organic Chemistry	3	CHEM 2446 Organic Chemistry Lab	1
MATH 2110Q Multivariable Calculus	4	CHEM 2444 Organic Chemistry	3
CHEG 2103 Intro to Chem Engineering	3	MATH 2410Q Diff Equations	3
PHIL 1104 Ethics (Content Area 1) [1]	3	CHEG 2111 Thermodynamics I	3
		Diversity and Multiculture (Content Area 4) [1]	3
	17		**17**

JUNIOR YEAR

First Semester	*Credits*	*Second Semester*	*Credits*
CHEG 3112 Thermodynamics II	3	CHEG 3124 Heat & Mass Transfer	3
CHEG 3123 Fluid Mechanics	3	CHEG 3128 Junior Chem Engineering Lab	2
CHEG 3145 Chemical Engineering Analysis	3	CHEG 3151 Process Kinetics	3
Social Science (Content Area 2) [1]	3	Engineering Requirement [3]	3
MCB/Biology/CHEM Requirement [4]	4	Diversity and Multiculture (Content Area 4) [1]	3
		Free Elective	3
	16		**17**

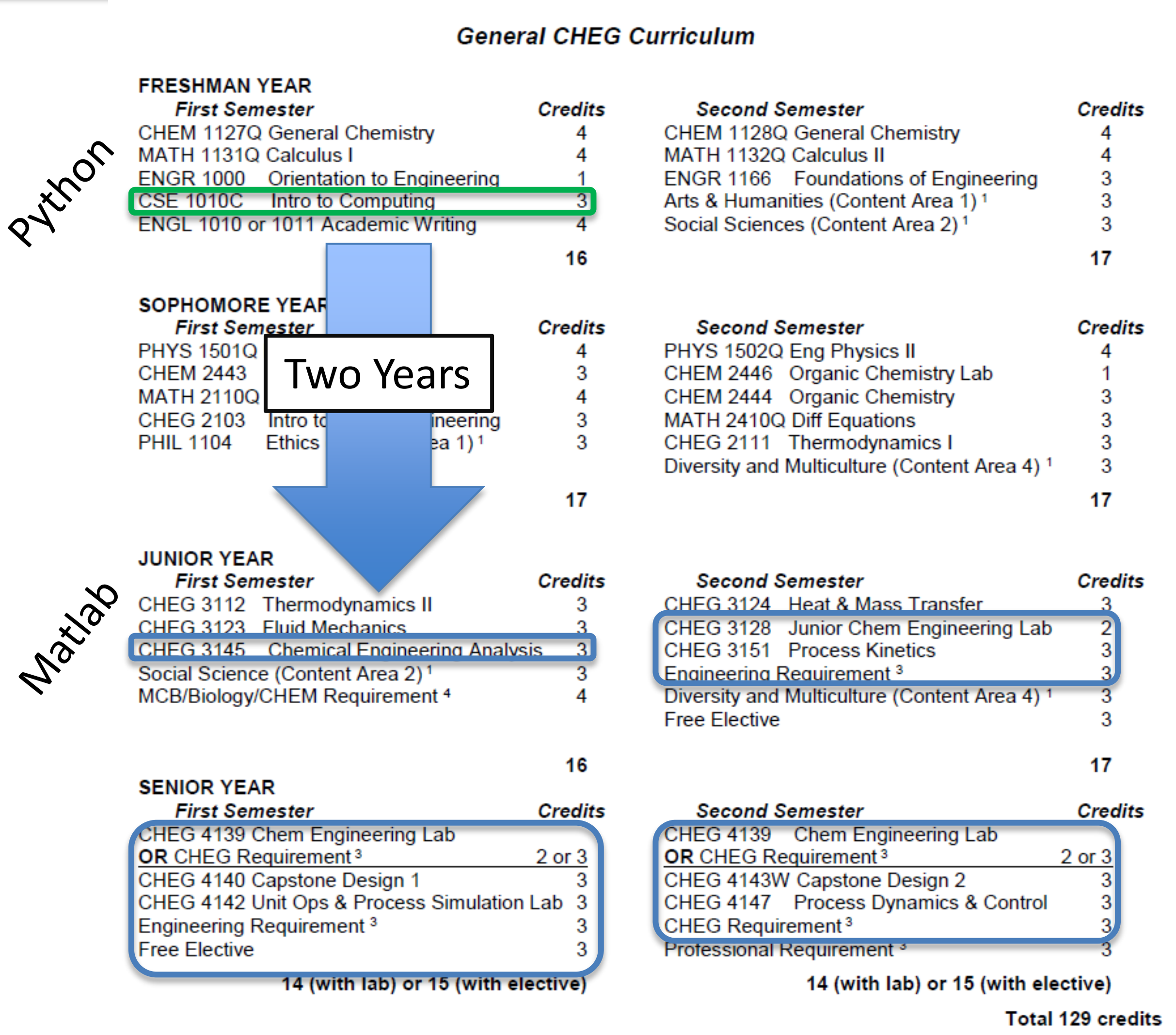
SENIOR YEAR

First Semester	*Credits*	*Second Semester*	*Credits*
CHEG 4139 Chem Engineering Lab **OR** CHEG Requirement [3]	2 or 3	CHEG 4139 Chem Engineering Lab **OR** CHEG Requirement [3]	2 or 3
CHEG 4140 Capstone Design 1	3	CHEG 4143W Capstone Design 2	3
CHEG 4142 Unit Ops & Process Simulation Lab	3	CHEG 4147 Process Dynamics & Control	3
Engineering Requirement [3]	3	CHEG Requirement [3]	3
Free Elective	3	Professional Requirement [3]	3
	14 (with lab) or 15 (with elective)		**14 (with lab) or 15 (with elective)**

Total 129 credits

Challenges: Content

- ✓ **Show students how to formulate and solve engineering problems**
- ✓ **Introduce concepts of algorithms**
- ✓ **Introduce numerical methods**
- ✓ **Teaching a "new" programming language**
  - *Intro to Computing* language determined by different department
  - 2 – 4 year gap after *Intro to Computing*
  - Interceding courses have minimal programming content



General CHEG Curriculum

FRESHMAN YEAR

First Semester	*Credits*	*Second Semester*	*Credits*
CHEM 1127Q General Chemistry	4	CHEM 1128Q General Chemistry	4
MATH 1131Q Calculus I	4	MATH 1132Q Calculus II	4
ENGR 1000 Orientation to Engineering	1	ENGR 1166 Foundations of Engineering	3
CSE 1010C Intro to Computing	3	Arts & Humanities (Content Area 1) [1]	3
ENGL 1010 or 1011 Academic Writing	4	Social Sciences (Content Area 2) [1]	3
	16		**17**

SOPHOMORE YEAR

First Semester	*Credits*	*Second Semester*	*Credits*
PHYS 1501Q	4	PHYS 1502Q Eng Physics II	4
CHEM 2443	3	CHEM 2446 Organic Chemistry Lab	1
MATH 2110Q	4	CHEM 2444 Organic Chemistry	3
CHEG 2103 Intro to ... ineering	3	MATH 2410Q Diff Equations	3
PHIL 1104 Ethics ... ea 1) [1]	3	CHEG 2111 Thermodynamics I	3
		Diversity and Multiculture (Content Area 4) [1]	3
	17		**17**

JUNIOR YEAR

First Semester	*Credits*	*Second Semester*	*Credits*
CHEG 3112 Thermodynamics II	3	CHEG 3124 Heat & Mass Transfer	3
CHEG 3123 Fluid Mechanics	3	CHEG 3128 Junior Chem Engineering Lab	2
CHEG 3145 Chemical Engineering Analysis	3	CHEG 3151 Process Kinetics	3
Social Science (Content Area 2) [1]	3	Engineering Requirement [3]	3
MCB/Biology/CHEM Requirement [4]	4	Diversity and Multiculture (Content Area 4) [1]	3
		Free Elective	3
	16		**17**

SENIOR YEAR

First Semester	*Credits*	*Second Semester*	*Credits*
CHEG 4139 Chem Engineering Lab **OR** CHEG Requirement [3]	2 or 3	CHEG 4139 Chem Engineering Lab **OR** CHEG Requirement [3]	2 or 3
CHEG 4140 Capstone Design 1	3	CHEG 4143W Capstone Design 2	3
CHEG 4142 Unit Ops & Process Simulation Lab	3	CHEG 4147 Process Dynamics & Control	3
Engineering Requirement [3]	3	CHEG Requirement [3]	3
Free Elective	3	Professional Requirement [3]	3
	14 (with lab) or 15 (with elective)		**14 (with lab) or 15 (with elective)**

Total 129 credits

Approach

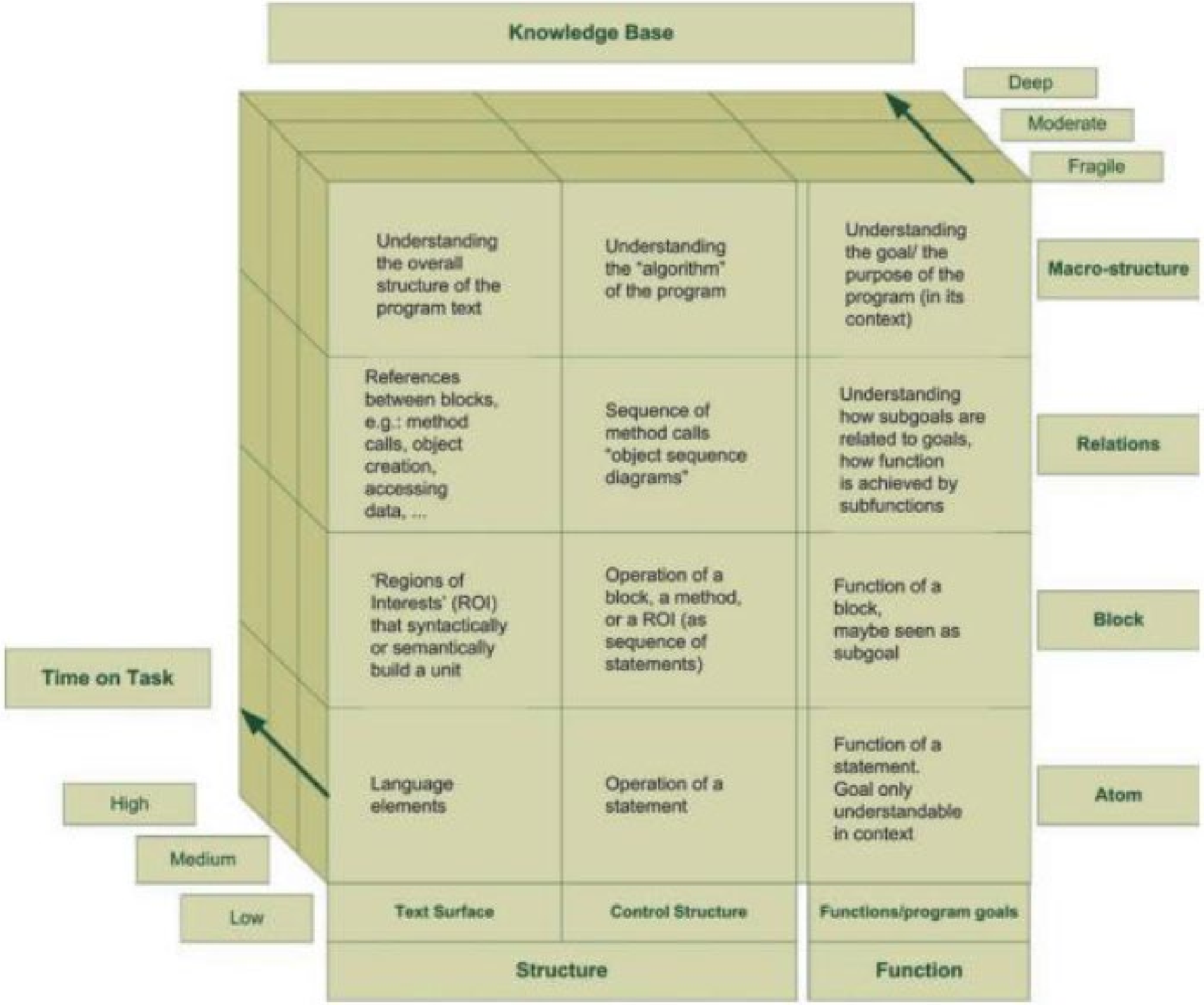
An integrated approach to improvement

New Interactive Modalities
- Student generated videos
- Interactive notebooks & assignments

Explicit Skills Focus
- Teach explicit coding skills and content
- Implicit comprehension monitoring

"Real World" Content
- Complex examples
- Literature and industry content

Content Gaps



Computational Thinking

- Problem formulation
- Recursion
- Problem Decomposition
- Abstraction
- Systematic testing
  - Debugging
  - Modular Programming

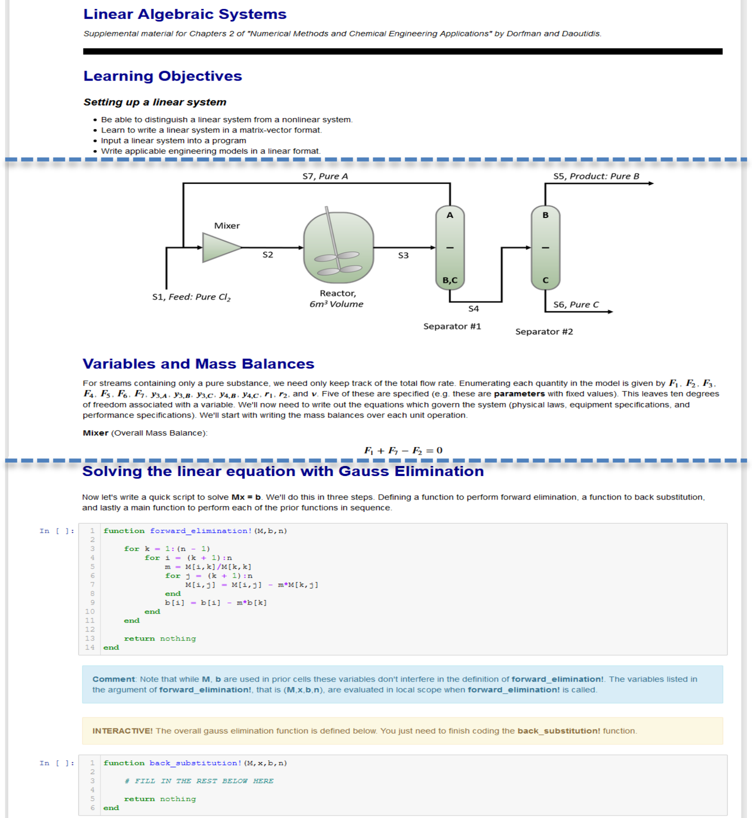
Valerie J. Shute, Chen Sun, and Jodi Asbell-Clarke. Demystifying computational thinking. Educational Research Review, 22:142–158, nov 2017. doi:10.1016/j.edurev.2017.09.003.

Jeannette M. Wing. Computational thinking. Communications of the ACM, 49(3):33, mar 2006. doi:10.1145/1118178.1118215.

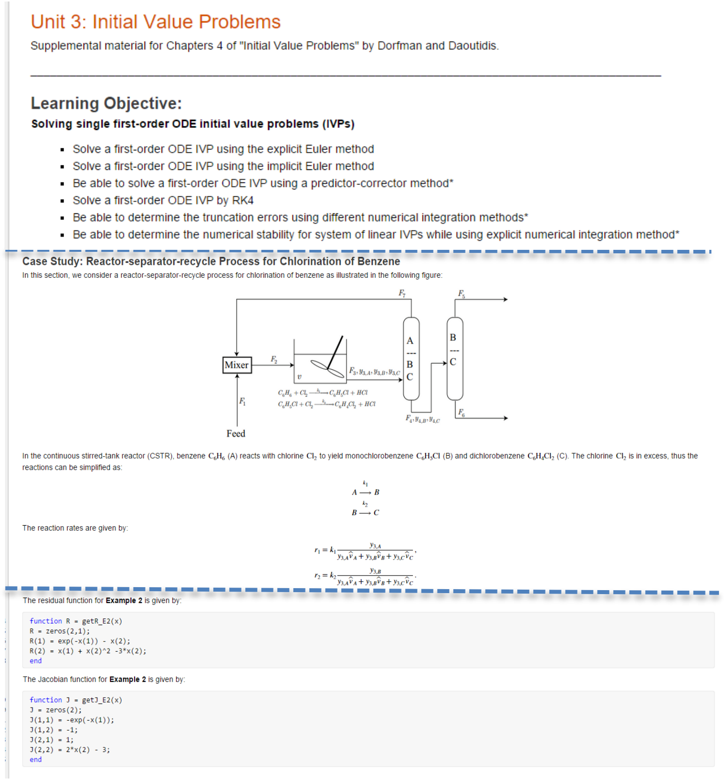
Notebook Design

- Designed notebooks for both Matlab (Live Editor) and Julia (Juptyer) initially
- A single document consisting of cells of either:
  - Rich text
  - Code
- Cells can be run individually or as a whole document

Jupyter Notebook:

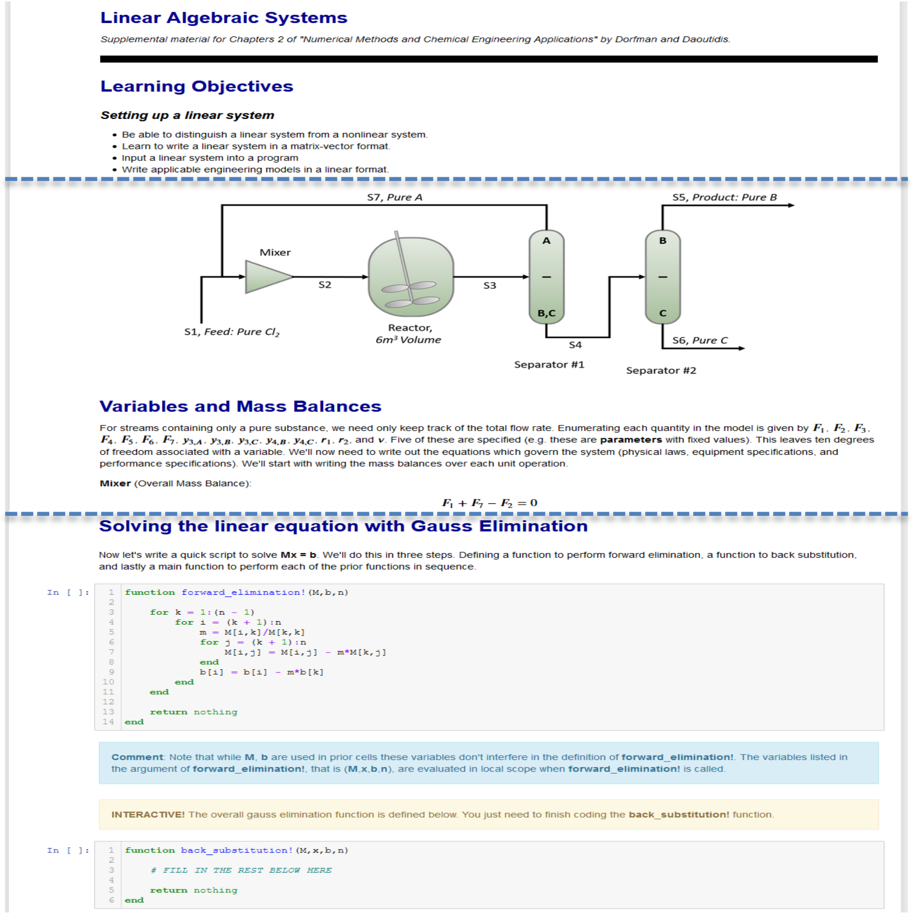


Matlab Live Editor:



Tutorial Design #1

- **Statement of unit level objectives**
- **Brief Review of Theory**
  - Links to website detailing industry applications and relevant academic research.
  - Comments on extension material.
- **1 – 2 Problems**
  - Background statements to frame a problem in an industrial context.
  - Interactive content.
  - Commentary on coding fundamental interlaced throughout the example.
  - Prompts for reflection interspersed.
- **End with reflective questions.**



Linear Algebraic Systems

Supplemental material for Chapters 2 of "Numerical Methods and Chemical Engineering Applications" by Dorfman and Daoutidis.

Learning Objectives

Setting up a linear system

- Be able to distinguish a linear system from a nonlinear system.
- Learn to write a linear system in a matrix-vector format.
- Input a linear system into a program
- Write applicable engineering models in a linear format.

Variables and Mass Balances

For streams containing only a pure substance, we need only keep track of the total flow rate. Enumerating each quantity in the model is given by F_1, F_2, F_3, F_4, F_5, F_6, F_7, $y_{3,A}$, $y_{3,B}$, $y_{3,C}$, $y_{4,B}$, $y_{4,C}$, r_1, r_2, and v. Five of these are specified (e.g. these are **parameters** with fixed values). This leaves ten degrees of freedom associated with a variable. We'll now need to write out the equations which govern the system (physical laws, equipment specifications, and performance specifications). We'll start with writing the mass balances over each unit operation.

Mixer (Overall Mass Balance):

$$F_1 + F_7 - F_2 = 0$$

Solving the linear equation with Gauss Elimination

Now let's write a quick script to solve **Mx = b**. We'll do this in three steps. Defining a function to perform forward elimination, a function to back substitution, and lastly a main function to perform each of the prior functions in sequence.

In [ ]:

```
function forward_elimination!(M,b,n)

    for k = 1:(n - 1)
        for i = (k + 1):n
            m = M[i,k]/M[k,k]
            for j = (k + 1):n
                M[i,j] = M[i,j] - m*M[k,j]
            end
            b[i] = b[i] - m*b[k]
        end
    end

    return nothing
end
```

Comment: Note that while **M**, **b** are used in prior cells these variables don't interfere in the definition of **forward_elimination!**. The variables listed in the argument of **forward_elimination!**, that is (**M**,**x**,**b**,**n**), are evaluated in local scope when **forward_elimination!** is called.

INTERACTIVE! The overall gauss elimination function is defined below. You just need to finish coding the **back_substitution!** function.

In [ ]:

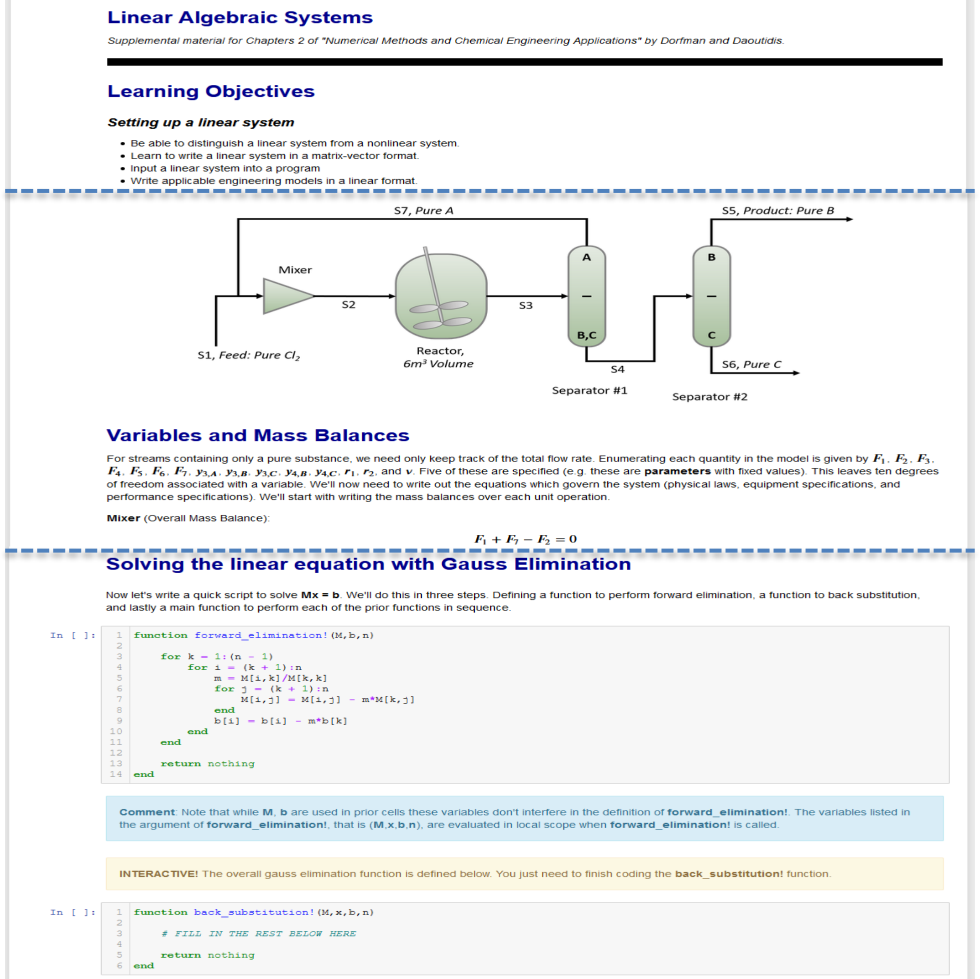
```
function back_substitution!(M,x,b,n)

    # FILL IN THE REST BELOW HERE

    return nothing
end
```

Jupyter Notebooks

- Supports 100+ languages: Matlab, Julia, Python, etc.
- **Containerization support for open-source languages: Both free and commercial support (Docker, MyBinder, etc).**
- **More features via extensions**
  - Hyperlinking to videos
  - Richer text formatting options
- **Simple installation for open-source languages**
- **Difficult installation with Matlab**
  - Install python -> Add/Update Packages Manually -> Change Environmental Variables
  - OS specific differences
- **Minimal formatting restrictions.**



Linear Algebraic Systems

Supplemental material for Chapters 2 of "Numerical Methods and Chemical Engineering Applications" by Dorfman and Daoutidis.

Learning Objectives

Setting up a linear system

- Be able to distinguish a linear system from a nonlinear system.
- Learn to write a linear system in a matrix-vector format.
- Input a linear system into a program
- Write applicable engineering models in a linear format.

Variables and Mass Balances

For streams containing only a pure substance, we need only keep track of the total flow rate. Enumerating each quantity in the model is given by F_1, F_2, F_3, F_4, F_5, F_6, F_7, $y_{3,A}$, $y_{3,B}$, $y_{3,C}$, $y_{4,B}$, $y_{4,C}$, r_1, r_2, and v. Five of these are specified (e.g. these are **parameters** with fixed values). This leaves ten degrees of freedom associated with a variable. We'll now need to write out the equations which govern the system (physical laws, equipment specifications, and performance specifications). We'll start with writing the mass balances over each unit operation.

Mixer (Overall Mass Balance):

$$F_1 + F_7 - F_2 = 0$$

Solving the linear equation with Gauss Elimination

Now let's write a quick script to solve **Mx = b**. We'll do this in three steps. Defining a function to perform forward elimination, a function to back substitution, and lastly a main function to perform each of the prior functions in sequence.

In [ ]:

```
function forward_elimination!(M,b,n)

    for k = 1:(n - 1)
        for i = (k + 1):n
            m = M[i,k]/M[k,k]
            for j = (k + 1):n
                M[i,j] = M[i,j] - m*M[k,j]
            end
            b[i] = b[i] - m*b[k]
        end
    end

    return nothing
end
```

Comment: Note that while **M**, **b** are used in prior cells these variables don't interfere in the definition of **forward_elimination!**. The variables listed in the argument of **forward_elimination!**, that is (**M**,**x**,**b**,**n**), are evaluated in local scope when **forward_elimination!** is called.

INTERACTIVE! The overall gauss elimination function is defined below. You just need to finish coding the **back_substitution!** function.

In [ ]:

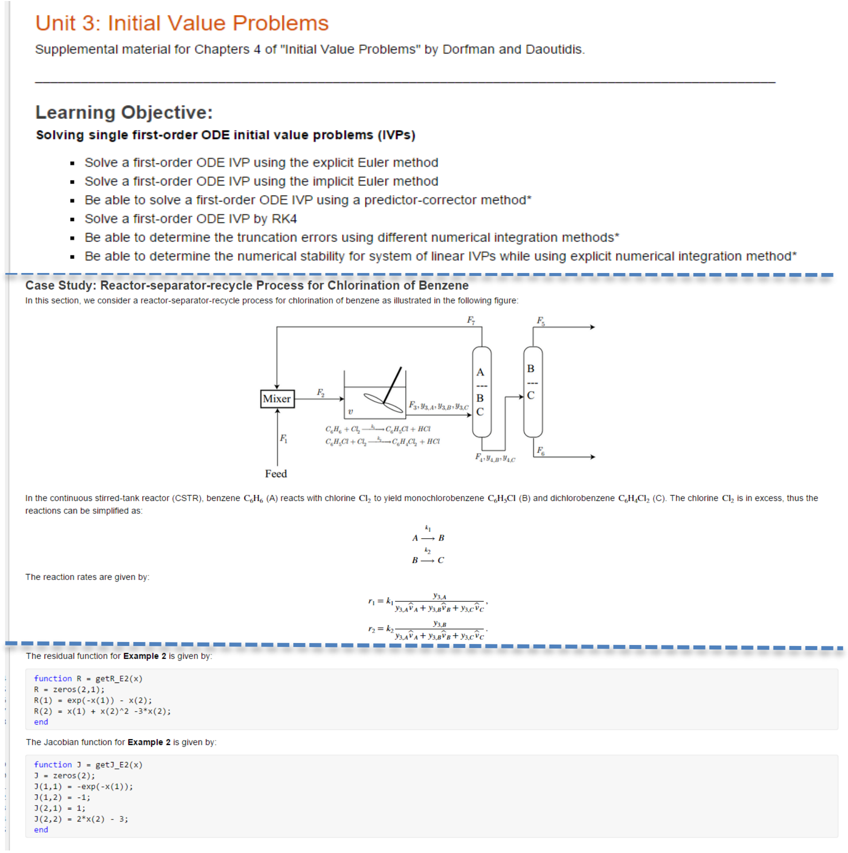
```
function back_substitution!(M,x,b,n)

    # FILL IN THE REST BELOW HERE

    return nothing
end
```

Matlab Live Editor

- **Available with Matlab Installation.**
- **Limited support for Rich text:**
  - Latex, Images, Figures
  - Not as readily extendible, no HTML support, embedding videos is tricky.
- **Live functions need to be placed at the bottom of the page.**
  - Modular programming in this environment must break the flow.
  - Can't have students ***effectively*** introduce functions in intermediate cells.



Unit 3: Initial Value Problems

Supplemental material for Chapters 4 of "Initial Value Problems" by Dorfman and Daoutidis.

Learning Objective:

Solving single first-order ODE initial value problems (IVPs)

- Solve a first-order ODE IVP using the explicit Euler method
- Solve a first-order ODE IVP using the implicit Euler method
- Be able to solve a first-order ODE IVP using a predictor-corrector method*
- Solve a first-order ODE IVP by RK4
- Be able to determine the truncation errors using different numerical integration methods*
- Be able to determine the numerical stability for system of linear IVPs while using explicit numerical integration method*

Case Study: Reactor-separator-recycle Process for Chlorination of Benzene

In this section, we consider a reactor-separator-recycle process for chlorination of benzene as illustrated in the following figure:

In the continuous stirred-tank reactor (CSTR), benzene C_6H_6 (A) reacts with chlorine Cl_2 to yield monochlorobenzene C_6H_5Cl (B) and dichlorobenzene $C_6H_4Cl_2$ (C). The chlorine Cl_2 is in excess, thus the reactions can be simplified as:

$$A \xrightarrow{k_1} B$$
$$B \xrightarrow{k_2} C$$

The reaction rates are given by:

$$r_1 = k_1 \frac{y_{3,A}}{y_{3,A}\hat{v}_A + y_{3,B}\hat{v}_B + y_{3,C}\hat{v}_C},$$
$$r_2 = k_2 \frac{y_{3,B}}{y_{3,A}\hat{v}_A + y_{3,B}\hat{v}_B + y_{3,C}\hat{v}_C}.$$

The residual function for **Example 2** is given by:

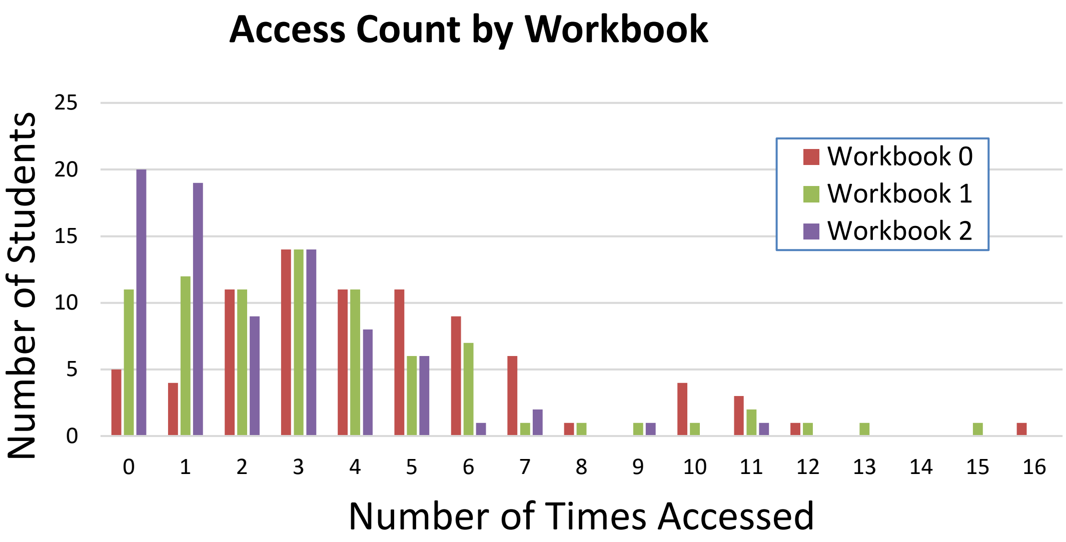
```
function R = getR_E2(x)
R = zeros(2,1);
R(1) = exp(-x(1)) - x(2);
R(2) = x(1) + x(2)^2 -3*x(2);
end
```

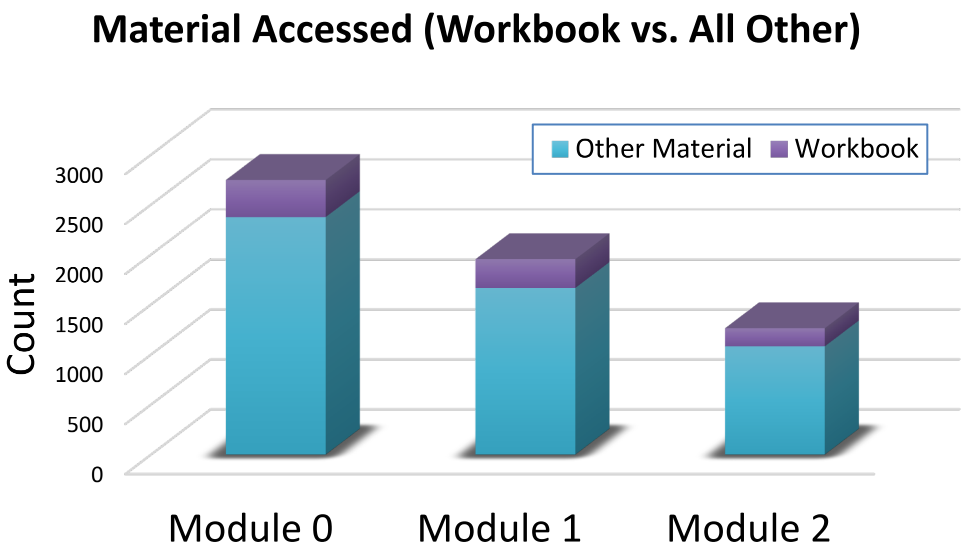
The Jacobian function for **Example 2** is given by:

```
function J = getJ_E2(x)
J = zeros(2);
J(1,1) = -exp(-x(1));
J(1,2) = -1;
J(2,1) = 1;
J(2,2) = 2*x(2) - 3;
end
```

Notebooks – Preliminary Data, Usage

- Interactive notebooks have been distributed this Fall semester as supplemental material.
- Usage information is encouraging (15% of access to content, nearly all students, multiple uses).





Notebooks – Preliminary Data, Performance

- We take a preliminary look at correlation between interactive notebook usage and examination scores for the first two exams.
- Equal frequency binning applied based on interaction number
- We see that low usage of the notebooks correspond to with lowest exam performance in each case.

Interactions vs. Exam 1	
Interaction	**Average Score (n)**
LOW (<= 5)	65.8% (24)
MED (5 to 9)	71.9% (25)
HIGH (> 9)	71.5% (24)

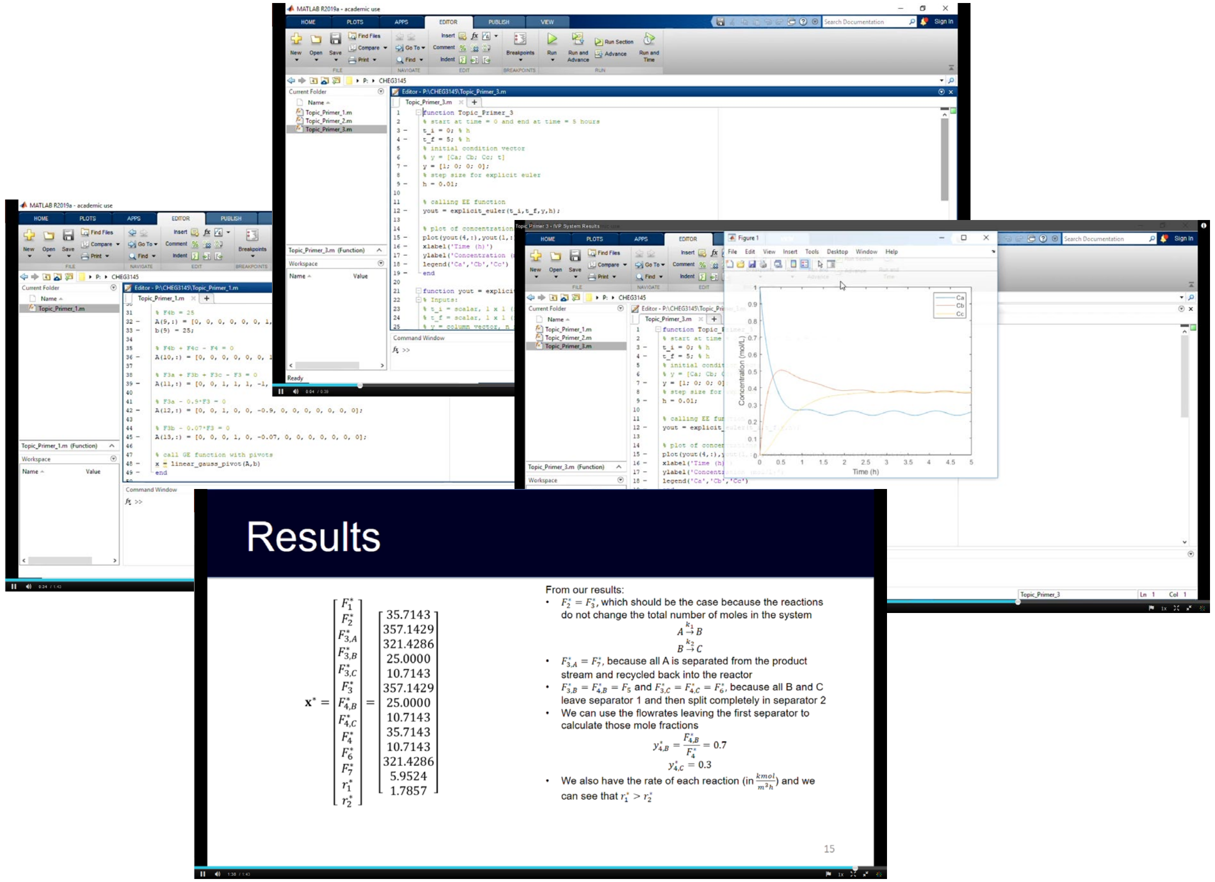
Interactions vs. Exam 2	
Interaction	**Average Score (n)**
LOW (<= 2)	61.7% (28)
MED (3 or 4)	77.3% (24)
HIGH (> 4)	70.7% (21)

Student Generated Videos

- Undergraduate teaching assistants generated short topic primer videos (1 – 5 minutes). Scripted developed by student and workshopped prior to recording content.

- Contains an example in which a student solves a sample problem while providing exposition.

- Content driven slides are interspersed. This helps highlight the role of problem formulation and abstraction-based thinking while modelling an approach rooted in computational thinking.



Student Videos – Preliminary Data

Final Grades by Interactions w/ Video Content				
Number of Interactions	***2017***	***2018***	***2019***	***2020***
No Interaction	62.5%	66.4%	64.0%	N/A*
One Video	N/A	N/A	70.4%	N/A*
Both Videos	N/A	N/A	74.8%	N/A*
All	N/A	N/A	72.1%	N/A*

* Course re-organized to a learning module format due to anticipated online delivery. Interaction with both videos increased to 96% (4.4 views on average).

Equal frequency binning applied based on interactions with Videos

Interactions vs. Exam	
Interaction	**Average Score (n)**
LOW (<= 6)	61.7% (25)
MED (7 to 10)	77.3% (24)
HIGH (> 10)	70.7% (24)

Next Steps

Initial Design	Instructional videos have been recorded and preliminary version of the interactive notebooks were constructed.
Preliminary Trial	Instructional videos and literate programming notebooks have been included as supplemental material within existing learning modules for this Fall semester.
Feedback	Usage statistics are currently being monitored and a preliminary survey will collect student feedback pertaining to the interactive notebooks.
Finalize	Interactive notebooks will be updated based on student feedback
Distribute	Interactive notebooks of both the Matlab Live Editor® and Jupyter notebooks formats will be distributed through the [CACHE] organization.
Rollout	Interactive notebooks will become required material and completion thereof will be linked to assessment.

Acknowledgements

Undergraduate Teaching Assistants - Notebook Editing

- Megan Walsh
- Talha Bhatti

Undergraduate Research Assistant – Video Content

- Dimitri Alston

Funding

Provided by grants from the Center for Excellence in Teaching and Learning at the University of Connecticut (UCONN CETL) and Computer Aids for Chemical Engineering (CACHE)

Computer Aids for Chemical Engineering

UCONN
UNIVERSITY OF CONNECTICUT
Center for Excellence in Teaching and Learning